Modul 323 – Funktional Programmieren

Vorwort:
Grundlage für diese Aufgabe ist die Präsentation «Einführung HTML & CSS»

Dauer: 90 Minuten
Optional: 0 Minuten
Total: 90 Minuten

Nachfolgend werden Sie sich mit den Technologien HTML und CSS auseinandersetzen. Diese werden im Verlaufe des Kurses dafür eingesetzt, um den Übungen ein Benutzerschnittstelle zu verleihen.

💡 **Der aus diesem Arbeitsblatt entstehende Source Code soll in ein Github-Repository eingecheckt werden und der Link in die folgende Text Box geschrieben werden:**
ℹ️ Tipp: Erstellen Sie eine «.gitignore» Datei und fügen Sie dieser «node_modules» hinzu. Die Module sollten NIE in ein git-Repository eingecheckt werden.

https://github.com/Kaenu/00_htmlcss_mod323

Aufgabe 1)
In dieser Aufgabe beschäftigen Sie sich mit dem Grundaufbau einer NodeJS Applikation, welche Sie für spätere Arbeitsblätter und Übungen verwenden können.

Erstellen Sie einen Ordner und Kopieren Sie die Dateien «package.json» und «webpack.config.js» aus «04_Data» in diesen Ordner. Installieren Sie danach die Pakete aus der «package.json» Datei. Verwenden Sie dazu «npm» oder «yarn».
ℹ️ Tipp: yarn kann über npm installiert werden «npm install --global yarn»

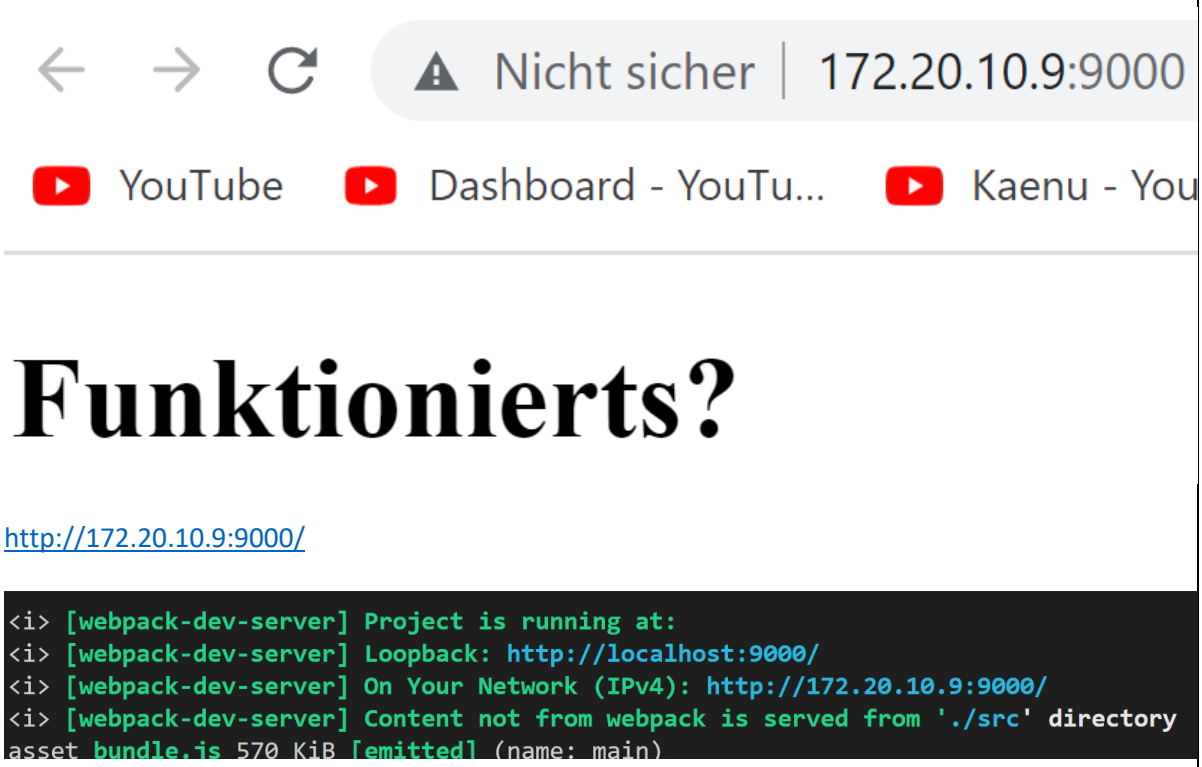
npm install

Erstellen Sie in dem Projektordner einen Ordner mit dem Namen «src». Erstellen Sie wiederum in diesem Ordner zwei Dateien mit dem Namen «index.html» und «index.js». Fügen Sie in der HTML-Datei ein h1 Titel ein. Starten Sie danach den Entwicklungsserver. Überprüfen Sie, ob Sie den h1 Titel sehen, wenn Sie auf den Entwicklungsserver zugreifen. Über welche Adresse kann der Entwicklungsserver aufgerufen werden?
ℹ️ Tipp: «yarn dev oder npm run dev»

Inhalt von index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
```

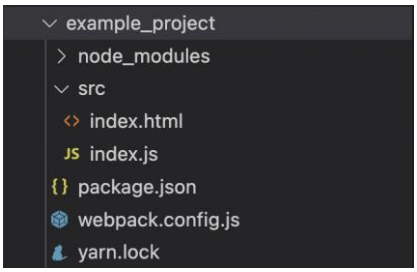
```
  <h1>Funktionierts?</h1>
</body>
</html>
```



http://172.20.10.9:9000/

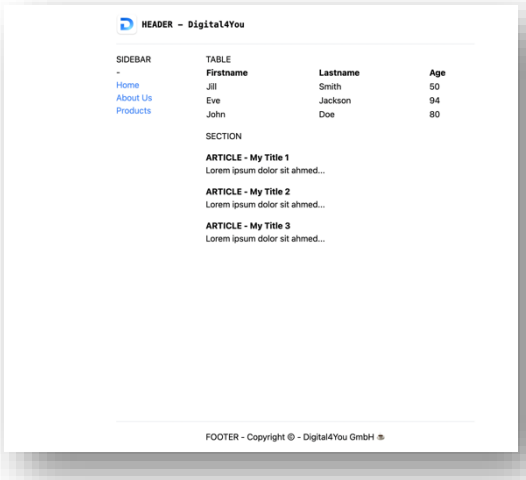
```
<i> [webpack-dev-server] Project is running at:
<i> [webpack-dev-server] Loopback: http://localhost:9000/
<i> [webpack-dev-server] On Your Network (IPv4): http://172.20.10.9:9000/
<i> [webpack-dev-server] Content not from webpack is served from './src' directory
asset bundle.js 570 KiB [emitted] (name: main)
```

Der Grundaufbau könnte nachher ungefähr folgendermassen aussehen.



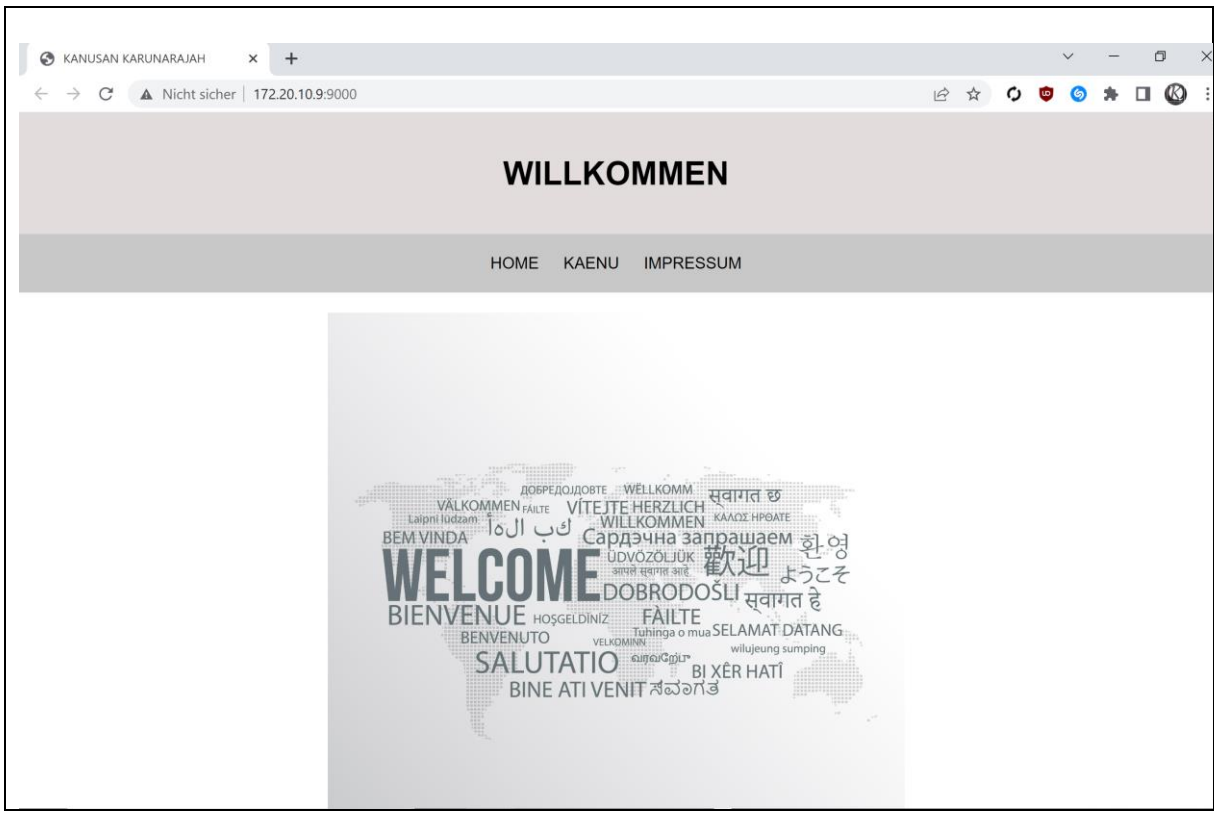
Aufgabe 2)

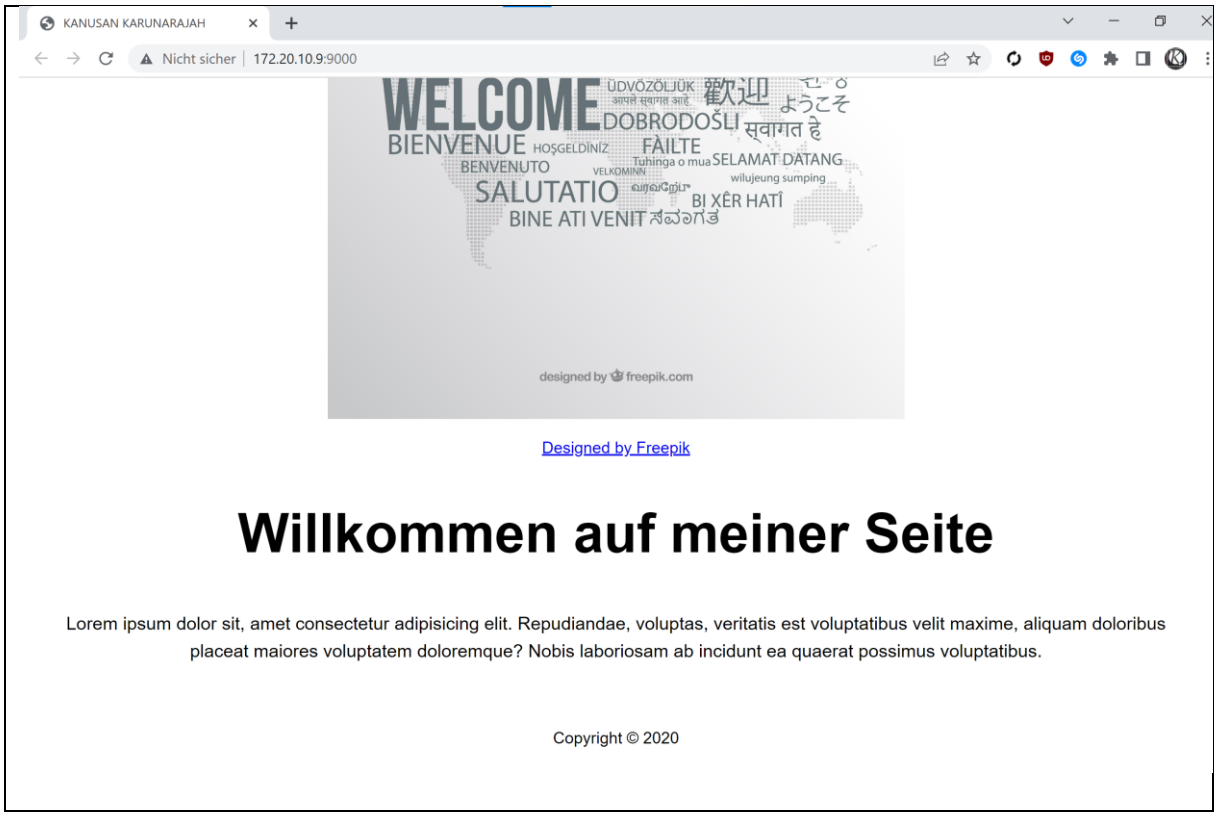
Setzen Sie sich nun mit dem Aufbau von HTML und CSS auseinander und erstellen Sie eine Grundstruktur mit entsprechenden HTML-5 Tags. Da Sie bereits Module hatten in denen HTML und CSS ein Thema war, wird davon ausgegangen, dass Sie diese in den Grundzügen beherrschen und es wird nicht noch einmal von Grund auf alles repetiert.



Erstellen Sie mit HTML 5 Tags eine Grundstruktur damit sie folgenden Aufbau erreichen.
Dabei soll das Layout mit eigenen CSS-Klassen umgesetzt werden. Diese sollen in einer externen CSS-Datei definiert werden. Füllen Sie die Sektionen mit für Sie (einigermassen) sinnvollem Inhalt wie Bilder, Navigationspunkten, Titelbild, Copyright usw.

ℹ️ Tipp (falls Sie nicht mehr genau wissen, wie Sie eine externe CSS-Datei einbinden können): https://www.w3schools.com/css/css_howto.asp





Aufgabe 3)
Sie haben das Layout zuvor mit dem klassischen Verfahren umgesetzt. Das bedeutet, Sie haben die CSS-Klassen selbst geschrieben. Wir verwenden für die zukünftigen Übungen eine etwas modernere Variante. Wir werden Tailwind CSS als «utility-classes» CSS-Framework verwenden, sodass Sie nicht mehr selbst CSS schreiben müssen.

Um die Entwicklung etwas angenehmer zu machen, gibt es für Tailwind CSS ein Visual Studio Code Plugin. Das Plugin heisst «Tailwind CSS IntelliSense». Erstellen Sie nach der Installation des Plugins eine leere Datei im «src» Ordner mit dem Namen «tailwind.config.js». Dies wird das Plugin aktivieren. Was genau ermöglicht dieses Plugin nun genau?
ℹ️ Tipp (Einbinden von Tailwind CSS ins HTML über ein CDN):
«<script src="https://cdn.tailwindcss.com"></script>»

Tailwind CSS IntelliSense ermöglicht Entwicklern, die Visual Studio Code verwenden, eine bessere Autovervollständigung, um Tailwind CSS-Klassen schnell und effizient einzugeben. Es bietet eine intelligente Autovervollständigung, Vorschläge für Responsive-Varianten, Farbvorschläge, bessere Erkennung von Verweisen und mehr

Setzen Sie das gleiche Layout wie bei Aufgabe 2 noch einmal um, verwenden Sie diesmal aber ausschliesslich Tailwind CSS «utility-classes» und erstellen Sie keine eigenen CSS-Klassen.
ℹ️ Tipp (verwenden Sie die offizielle Dokumentation oder googeln Sie. Zu diesem Thema werden Sie sehr viel finden, sofern Sie in englischer Sprache suchen): https://tailwindcss.com/docs

tailwind.config.js:

```
module.exports = {
  theme: {
    colors: {
      'gray-700': '#c2b7b778',
      'gray-200': '#c8c8c8'
    },
    fontFamily: {
      sans: 'sans-serif'
    },
    fontSize: {
      '2xl': '36px',
      'base': '18px'
    },
    lineHeight: {
      'normal': '1.5'
    },
    spacing: {
      '4': '20px',
      '10': '10px'
    }
  },
  variants: {},
  plugins: []
}
```

Aufgabe 4) (Optional)

Falls Sie mit dem Arbeitsblatt bereits fertig sind, können Sie sich den nachfolgenden Onlinekurs anschauen und selbstständig anfangen die Übungen zu erledigen. Das kann Ihnen zusätzlich helfen, in das ganze Thema hereinzukommen.

https://www.freecodecamp.org/news/functional-programming-in-javascript-for-beginners/

💡 Um die Übungen und den Fortschritt speichern zu können, müssen Sie einen gratis Account auf freecodecamp erstellen.